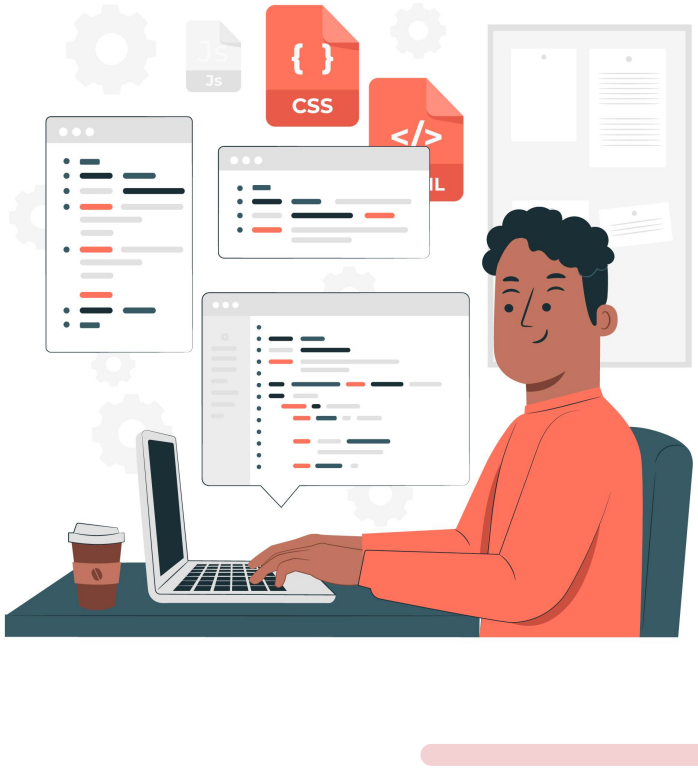
KEMNAKER

Sanbercode

Modern JavaScript: Functional, Async, ES6+ & OOP



KEMNAKER

sanbercode

Outline

- Functional Programming
- ES6+

KEMNAKER

Sanbercode

Functional Programming

KEMNAKER

Sanbercode

Functional Programming

Kita sering mendengar berbagai paradigma pemrograman seperti object-oriented, procedural, dan functional. JavaScript adalah bahasa pemograman multi-paradigma yang memungkinkan kita untuk secara bebas mencampur dan mencocokkan paradigma pemograman.

Apa itu functional programming? yaitu sebuah paradigma dalam dunia programming yang menggunakan pendekatan fungsional untuk menyelesaikan masalah. Apakah jika kode program kita menggunakan function maka itu pasti disebut sebagai functional programming? ternyata tidak, paradigma ini mensyaratkan penggunaan pure function sebisa mungkin.

keuntungan Functional Programming

Berikut beberapa keuntungan yang kita dapatkan ketika menggunakan paradigma functional programming

- **Performa relatif lebih baik karena kode program menjadi relatif lebih singkat**
- **Kode program menjadi lebih terbaca**
- **Penggunaan pure function membuat kode program lebih mudah didebug**

Metode-Metode dalam Paradigma Functional Programming

Berikut ini beberapa metode-metode dalam paradigma functional programming

- **Recursive Function**
- **First-class Function**
- **Currying Function**
- **Declarative vs Imperative Pattern**

KEMNAKER

Recursive Function

Recursive function maksudnya adalah function yang memanggil dirinya sendiri sampai suatu kondisi pemanggilan dirinya berhenti. Contoh:

```
// program untuk menghitung mundur hingga angka 1
function countDown(number) {
  // menampilkan number
  console.log(number);

  // mengurangi nilai number
  var newNumber = number - 1;

  // kondisi jika nomor masih diatas 0
  if (newNumber > 0) {
      countDown(newNumber);
  }
}

countDown(4);
```

First-class Function

First-class function maksudnya adalah menjadikan fungsi layaknya class atau dalam hal ini kita bisa mendeklarasikan fungsi di dalam fungsi yang lain. Contoh:

```
function HitungLingkaran(radius) {
  var pi = 22/7

  function luas (r){return pi * r * r}
  function keliling (r){return 2 * pi * r }
  console.log("Luas:", luas(radius))
  console.log("Keliling:",keliling(radius))
}

HitungLingkaran(7)
```

Currying Function

Fungsi currying adalah teknik untuk mengubah bentuk suatu fungsi dengan beberapa argumen menjadi sebuah fungsi anak yang berurutan.
Fungsi currying akan mengembalikan fungsi baru yang sesuai dengan argumen berikutnya. Contoh:

```
function tambah (a, b){
  return a + b
}

console.log(tambah(2, 5)) // 7

// currying function
function tambah (a){
 return function(b){
   return a + b
 }
}

console.log(tambah(2)(5)) // 7
```

Apa Keunggulan Currying Function

pada contoh sebelumnya masih terasa Keunggulan dari currying function kecuali hanya untuk memberikan contoh bentuk dasarnya.

Teknik currying ini sangat berguna jika terdapat argumen dari fungsi itu yang membutuhkan komputasi berat

Contoh Komputasi berat tanpa currying function

```
// simulasi untuk komputasi berat
function getGreeting(){
    var x = 0
    for (i=0; i<1000000000; i++){
        x += i
    }
    return "Selamat pagi"
}

function say(callback, name) {
    var greeting = callback()
    console.log(greeting, name)
}

var users = ['Fuad', 'Hasan', 'Ismi', 'Budi', 'Azhar']

users.map(function(user) {
  return say(getGreeting, user)
})
```

KEMNAKER

Contoh Komputasi berat dengan currying function

```
// simulasi untuk komputasi berat
function getGreeting(){
  var x = 0
  for (i=0; i<1000000000; i++){
      x += i
  }
  return "Selamat pagi"
}

function say(callback) {
  var greeting = callback()
  return function(name){
    console.log(greeting, name)
  }
}

var users = ['Fuad', 'Hasan', 'Ismi', 'Budi', 'Azhar']
var sayGretting = say(getGreeting)

users.map(function(user) {
  return sayGreeting(user)
})
```

Declarative vs Imperative Pattern

Terdapat dua pattern atau pola ketika membuat kode program yaitu:

- **Imperative berfokus pada gambaran bagaimana suatu program beroperasi, ada step by stepnya.**
- **Declarative berfokus pada apa yang harus dicapai oleh program tanpa menjabarkan step by stepnya.**

Imperative Pattern

berikut ini contoh imperative pattern:

```
// Imperative: Bagaimana menghitung jumlah bilangan genap dalam array
function hitungBilanganGenap(arr) {
  var count = 0;
  for (var i = 0; i < arr.length; i++) {
    if (arr[i] % 2 === 0) {
      count++;
    }
  }
  return count;
}

var angka = [1, 2, 3, 4, 5, 6];
var jumlahGenap = hitungBilanganGenap(angka);
console.log(jumlahGenap); // Output: 3
```

Declarative Pattern

berikut ini contoh declarative pattern:

```
// Declarative: Apa yang ingin dicapai
var angka = [1, 2, 3, 4, 5, 6];
var jumlahGenap = angka.filter(function(x){return x % 2 === 0}).length;
console.log(jumlahGenap); // Output: 3
```

KEMNAKER

Sanbercode

Sesi Tanya Jawab

KEMNAKER

Sanbercode

ES6+

Apa itu ES6

ES6 atau EcmaScript 6 atau EcmaScript 2015 merupakan fitur bahasa pada pemrograman Javascript modern atau biasa di sebut next gen javascript. lalu kenapa kita harus mempelajari ES6, itu karena library maupun framework JS terkini banyak menggunakan sintaks-sintaks ES6.

setelah EcmaScript 6 ini banyak fitur-fitur yang dapat digunakan untuk mengoptimalkan

KEMNAKER

Sanbercode

Daftar Fitur ES6

berikut ini daftar fitur ES6:

- Arrows Function
- Classes
- Enchanced object literals
- Template strings
- Destructuring
- Default + rest + spread
- Let + const
- Iterators + for..of
- Generators

KEMNAKER

Sanbercode

Apa Perlu Menguasai Seluruh Fitur dari awal?

Tidak perlu menguasai seluruh fitur diatas, tetapi kita akan membagi fitur-fitur yang penting untuk di pelajari yaitu:

- let + const
- arrow function
- default paramater
- Template literal
- Enhanced object literals
- Destructuring
- Rest Parameters + Spread Operator
- promise
  (akan dibahas pada materi asynchronous)
- class
  (akan dibahas pada materi OOP)

KEMNAKER

Sanbercode

Let + Const

let dan const merupakan statement untuk mendefinisikan variable sama seperti var hanya saja terdapat perbedaan diantara let + const dan var.

let hampir sama seperti var hanya saja jika dalam satu block kode ada sebuah let yang sudah di definisikan kita tidak dapat mendefinisikan kembali kecuali kita membuat block kode baru di dalamnya misal seperti menambahkan if. sedangkan const sendiri bersifat tidak bisa di rubah seperti sebuah konstanta dalam matematika. var sendiri masih bisa digunakan untuk menjaga compability dengan versi sebelumnya

Arrow Functions

arrow functions merupakan fitur yang ada pada es6 bisa dibilang lebih singkat dari function biasa, function biasa sendiri masih bisa di gunakan.

contoh sintaks:

```
const myFunc = ()=>{}
```

Default Parameter

biasanya kita dalam membuat function pasti punya parameter tapi apakah parameter itu sendiri bisa di beri default

contoh sintaks:
function myFunc(param= default) {}

KEMNAKER

Sanbercode

Template Literal

template literal atau biasa di sebut template string merupakan fitur ES6 yang memungkinkan kita menyusun string dengan rapi dengan menggunakan tanda petik terbalik dan ${variabelnya}.

contoh sintaks:

```
let word = “example”
let result = `this is ${word}`
```

Enhanced object literals

Enhanced object literals merupakan fitur ES6 yang memungkinkan kita untuk menyederhanakan sebuah object, dimana biasanya kita selalu menulis property dan value, tetapi jika terdapat kondisi ada variabel yang namanya sama dengan property maka kita bisa assign hal tersebut sebagai value tetapi dengan hanya menulis property nya saja

Destructuring

Destructuring merupakan ekspresi javascript yang memungkinkan untuk membagi atau memecah nilai dari sebuah array atau objek ke dalam variabel yang berbeda

KEMNAKER

Sanbercode

Rest Parameter + Spread Operator

Rest Parameters dan Spread Operator di lambangkan dengan simbol yang sama yaitu "..."

KEMNAKER

Sanbercode

Rest Parameter

Rest Parameter ini berguna untuk menggabungkan semua paramater pada function ke dalam array. Dengan menggunakan Rest Parameter ini dapat membantu kita mendefinisikan function dengan rapi serta memberikan parameter yang tidak terbatas pada sebuah function.

Spread Operator

Spread Operator digunakan untuk membagi elemen array atau properti pada objek, sehingga elemen array dapat ditambahkan/dimasukan ke dalam array baru

KEMNAKER

sanbercode

Kesimpulan

- Functional Programming
- ES6+

KEMNAKER

Sanbercode

Terima Kasih